برنامه‌سازی پیشرفته

تمرین کامپیوتری شماره ۳

مدرس: رامتین خسروی

طراحان: آرتین توسلی، منا شهرکی، حمید محمودی، متین رزاقی‌زاده، کوثر شیری، پریا پاسه‌ورز، مانی حسینی، محمدرضا یعقوبی، عرفان فلاحتی، نیما ادیب

مهلت تحویل: شنبه ۱۹ اردیبهشت ۱۴۰۵، ساعت ۲۳:۵۹

مقدمه

هدف از این تمرین آشنایی شما با طراحی برنامه است. با توجه به حجم تمرین توصیه می‌شود این روال را برای انجام آن طی کنید.

1. قبل از شروع پیاده‌سازی زمانی را به خواندن کامل شرح مسئله اختصاص دهید.
2. پس از آن به نحوه شکست برنامه به اجزای کوچک‌تر (مثل روال‌های خواندن ورودی اولیه و رسیدگی به دستورات کاربر) فکر کنید.
3. بر اساس شکست بند قبل، درباره تایپ‌هایی که اطلاعات ثابت برنامه (در اینجا اطلاعات کامیون‌ها و مرسوله‌ها) را نگه می‌دارند تصمیم‌گیری کنید.
4. توصیه می‌شود به این ترتیب پیاده‌سازی خود را پیش ببرید:
   a. ابتدا خواندن فایل‌های ورودی را کامل کنید.
   b. دستورها را یکی یکی در برنامه خود پیاده‌سازی کنید.
   c. بعد از پیاده‌سازی هر دستور درستی کارکرد برنامه خود را با داده‌های آزمون بیازمایید.

   به این ترتیب، حتی اگر فرصت نکردید تمام برنامه را پیاده‌سازی کنید آنچه تا پایان مهلت تمرین دارید برنامه‌ای است که کار می‌کند (هر چند کامل نیست).

برای آشنایی بیشتر با مفاهیم طراحی بالا به پایین می‌توانید به ویدیویی که در بخش محتوای دستیاران آموزشی در صفحه درس درس مراجعه کنید.

یوتی تراک

این تمرین به پیاده‌سازی یک سیستم جامع مدیریت حمل‌ونقل بار می‌پردازد. در این سیستم، شبکه ارسال مرسولات به صورت متمرکز طراحی شده است؛ به این معنا که تمامی بارها صرفاً از مرکز اصلی (تهران) به شهرهای پیرامون ارسال شده و یا از این شهرها به تهران منتقل می‌شوند. هدف اصلی این برنامه، مدیریت هوشمند و یکپارچه فرایندها است که شامل دریافت اطلاعات پایه، تخصیص مناسب‌ترین کامیون به هر بار، رهگیری وضعیت مرسولات (از زمان بارگیری تا تحویل نهایی) و ارائه گزارش‌های دقیق عملکردی و مالی می‌شود.

نحوه اجرای برنامه

در زمان اجرای برنامه، مسیر فایل‌های ورودی به ترتیب (ابتدا مسیر فایل اطلاعات کامیون‌ها، سپس مسیر فایل اطلاعات شهرها) از طریق آرگومان خط فرمان به برنامه داده می‌شود. برای آشنایی با نحوه ارسال آرگومان‌های خط فرمان می‌توانید مستندات مربوط به آن را مطالعه کنید.

قالب ورودی

```
./UTTruck </path/to/trucks.csv> </path/to/cities.csv>
```

قالب فایل‌های ورودی

اطلاعات مربوط به کامیون‌ها و شهرها در دو فایل جداگانه قرار دارند. آدرس این فایل‌ها به صورت آرگومان خط فرمان به برنامه شما داده می‌شود. شما باید اطلاعات موجود در آن‌ها را در ابتدای اجرای برنامه خوانده و ذخیره کنید. نوع فایل‌های ورودی از نوع CSV هستند و برای آشنایی با این نوع فایل‌ها می‌توانید مستندات مربوط به فرمت CSV را بررسی کنید. هر فایل CSV یک سطر header در ابتدای خود دارد که نام ستون‌های فایل را مشخص می‌کند. در ادامه شرح هر یک از فایل‌ها آمده است.

فایل کامیون‌ها

در این فایل، برای هر کامیون دو ویژگی شناسه و ظرفیت ثبت شده است. تضمین می‌شود که شناسه هر کامیون یکتاست و ظرفیت آن (که نشان‌دهنده حداکثر وزن قابل‌حمل است) به‌صورت یک عدد طبیعی و بدون کاراکترِ فاصله[1] درج شده است.

نمونه فایل کامیون‌ها

```
truck_id,capacity
1,100
2,150
3,200
4,50
```

فایل شهرها

در این فایل، اطلاعات شهرها شامل نام شهر و فاصله آن تا تهران ثبت شده است. نام شهرها بدون کاراکترِ فاصله درج شده و فاصله آن‌ها تا تهران نیز به‌صورت یک عدد طبیعی مشخص می‌شود. همچنین، فاصله تهران تا خودش برابر با صفر در نظر گرفته شده است.

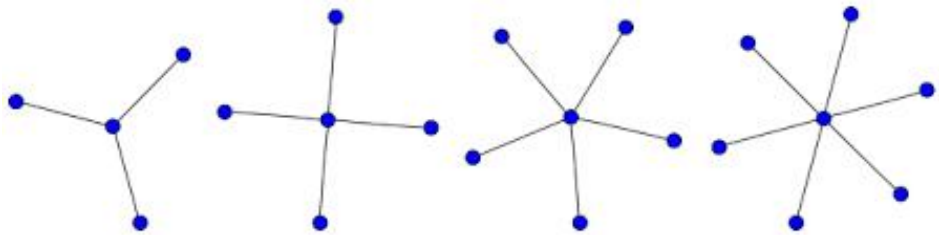
نمونه فایل شهرها

```
city_name,distance
Arak,500
Isfahan,700
```

[1] Space

ساختار سیستم حمل‌ونقل

در این ساختار تهران به عنوان شهر مرکزی قرار دارد و هر یک از شهرها تنها با یک مسیر مستقیم به تهران متصل هستند. هیچ مسیری بین شهرهای دیگر وجود ندارد. تمامی بارها صرفاً از مرکز اصلی (تهران) به شهرهای پیرامون ارسال شده و یا از این شهرها به تهران منتقل می‌شوند. در ابتدا، تمامی کامیون‌ها در تهران مستقر هستند و پس از تحویل بار، در همان شهرِ مقصد باقی می‌مانند. به عنوان نمونه، شکل زیر تعدادی از ساختارهای ممکن برای ارتباط شهرها با تهران را نمایش می‌دهد:



دستورات برنامه

دستور ثبت مرسوله جدید

با وارد کردن این دستور، یک مرسوله جدید با مبدا، مقصد و وزن مشخص در سیستم ثبت می‌شود. برای هر مرسوله یک شناسه یکتا اختصاص داده می‌شود که این شناسه از ۱ شروع شده و با ثبت هر مرسوله جدید به طور خودکار یک واحد افزایش پیدا می‌کند. این مرسوله در انبار قرار می‌گیرد تا در زمان مناسب توسط یکی از کامیون‌ها حمل شود. تضمین می‌شود که نام شهرهای وارد شده در سیستم تعریف شده‌اند، وزن داده‌شده عددی طبیعی است و مبدأ و مقصد یکسان نیستند. همچنین، همواره مبدأ یا مقصد هر مرسوله (دقیقاً یکی از آن‌ها) شهر تهران خواهد بود.

قالب ورودی
`add_order <origin_city> <destination_city> <weight>`
قالب خروجی
`Order <order_id> added`

نمونه ورودی
`add_order Tehran Arak 50`
نمونه خروجی
`Order 1 added`

دستور پیگیری وضعیت مرسوله

با وارد کردن این دستور می‌توان وضعیت فعلی یک مرسوله را مشاهده کرد. هر مرسوله در سیستم می‌تواند یکی از این سه وضعیت را داشته باشد: یا هنوز در انبار است، یا در مسیر، در حال حمل توسط کامیون است، و یا به مقصد نهایی تحویل داده شده است. زمانی که مرسوله ایجاد می‌شود، در ابتدا در انبار قرار دارد. در صورتی که مرسوله در انبار باشد باید نام شهر محل فعلی آن، و در صورتی که در حال حمل توسط کامیون باشد باید شهر مقصد کامیون در خروجی قابل مشاهده باشد. برای مرسوله‌های تحویل داده شده نیز شهر مقصد چاپ می‌شود.

اگر شناسه مرسوله وارد شده در سیستم وجود نداشته باشد، برنامه باید پیام `Order not found` را چاپ کند.

قالب ورودی
`track <order_id>`
قالب خروجی

`Order <order_id> is currently in warehouse in <city>`
`| Order <order_id> is in transit to <city>`
`| Order <order_id> is delivered to <city>`
`| Order not found`

نمونه ورودی
`track 1`
نمونه خروجی
`Order 1 is currently in warehouse in Tehran`

دستور بارگیری کامیون

با وارد کردن این دستور، کامیونی که شناسه آن وارد شده است برای مقصد مشخص بارگیری می‌شود. ابتدا تمام مرسوله‌هایی که مقصد آن‌ها همان مقصد وارد شده است و مبداشان با مبدا کامیون وارد شده یکی است، انتخاب می‌شوند. سپس برای هر مرسوله، امتیاز آن با توجه به وزن مرسوله و اختلاف شناسه آن با بزرگترین شناسه ثبت شده در سیستم محاسبه می‌شود:

$$score = weight + (LargestOrderID - OrderID) * 5$$

در صورتی که مرسوله ای برای ارسال به مقصد مورد نظر وجود نداشته باشد یا کامیون در مبدا نباشد، `No order could be loaded` باید چاپ شود.

مرسوله‌ها ابتدا بر اساس امتیاز (از بیشترین به کمترین) مرتب می‌شوند. سپس فرآیند بارگیری به ترتیب اولویت انجام می‌شود؛ مشروط بر اینکه وزن مرسوله انتخابی از ظرفیت باقی‌مانده کامیون بیشتر نباشد. پس از تکمیل بارگیری، کامیون بلافاصله اعزام شده و وضعیت آن به در حال حمل تغییر می‌کند. همچنین، تضمین می‌شود که شناسه کامیونِ وارد شده معتبر است. اگر امتیاز دو مرسوله یکسان بود، مرسوله‌ای که زودتر ثبت شده (شناسه کوچک‌تری دارد) در اولویت قرار می‌گیرد. شناسه‌ی مرسوله‌های بارگیری‌شده باید به ترتیب صعودی در خروجی چاپ شوند.

قالب ورودی
`load <truck_id> <destination_city>`
قالب خروجی
`Truck <truck_id> loaded with orders: <id1> <id2> ...` `| No order could be loaded`

نمونه ورودی
`load 1 Arak`
نمونه خروجی
`Truck 1 loaded with orders: 1 3 5`

دستور تحویل مرسوله

این دستور شناسه کامیون مورد نظر را ورودی می‌گیرد و تمام مرسوله‌های داخل آن را از در حال حمل به تحویل داده شده تغییر می‌دهد. تضمین می‌شود شناسه کامیون معتبر است. اگر کامیون خالی باشد و این دستور زده شود، پیام داده شده را چاپ کنید. شناسه‌ی مرسوله‌ها باید به ترتیب صعودی در خروجی چاپ شوند.

قالب ورودی
`deliver <truck_id>`
قالب خروجی
`Truck <truck_id> delivered orders: <o1 o2 ...>` `| No orders to deliver in truck <truck_id>`

نمونه ورودی
`deliver 1`
نمونه خروجی
`Truck 1 delivered orders: 1 3 5`

دستور پیشنهاد شهر و کامیون

برای انتخاب بهترین مقصد، ابتدا امتیاز مرسوله‌های ورودی و خروجی برای هر شهر محاسبه می‌شود. از آنجا که هدف سیستم ارسال هرچه سریع‌تر مرسوله‌های قدیمی‌تر و سنگین‌تر است، امتیاز هر مرسوله (مشابه دستور بارگیری) بر اساس رابطه زیر محاسبه می‌گردد:

$$OrderScore = Weight + (LargetOrderID - OrderID) * 5$$

سپس امتیاز هر شهر بر اساس مقادیر زیر حساب می‌شود:

- $SumOutgoingScores$ نشان‌دهنده مجموع امتیازات کالاهایی است که از شهر مقصد به تهران می‌روند .
- $SumIncomingScores$ نشان‌دهنده مجموع امتیازات کالاهایی است که از تهران به مقصد می‌روند.
- $distance(city)$ نیز فاصله تهران با شهر مقصد است.

$$CityScore = (SumOutgoingScores + SumIncomingScores) / distance(city)$$

شهری که بیشترین مقدار $CityScore$ را داشته باشد به عنوان مقصد پیشنهادی انتخاب می‌شود. در صورت برابر بودن مقدار $CityScore$ میان چند شهر، شهری که نام آن طبق ترتیب الفبایی در اولویت قرار می‌گیرد، به عنوان مقصد انتخاب می‌شود. پس از مشخص شدن شهر مقصد، برنامه از میان کامیون‌هایی که در حال حاضر در شهر تهران قرار دارند و ظرفیت آن‌ها برای حمل بارها کافی است، کامیونی را پیشنهاد می‌دهد که پس از بارگیری در تهران، کمترین ظرفیت خالی را داشته باشد. در صورتی که چندین کامیون کمترین ظرفیت خالی را داشتند، کامیون با کوچک‌ترین شناسه انتخاب می‌شود. تضمین می‌شود که حداقل یک شهر دارای مرسوله فعال و حداقل یک کامیون قابل استفاده وجود دارد.

قالب ورودی
`recommend`
قالب خروجی
`Recommended city: <city_name>` `Recommended truck: <truck_id>`

نمونه ورودی
recommend
نمونه خروجی
Recommended city: Arak Recommended truck: 4

دستور گزارش مالی

برای هر مرسوله یک هزینه بر حسب وزن و مسافت طی شده آن محاسبه می‌شود:

$$cost = distance * weight$$

این دستور مجموع درآمد شرکت را بر اساس مرسوله‌هایی که **تحویل داده شده‌اند** محاسبه و نمایش می‌دهد. برای هر مرسوله، هزینه‌ای که در زمان ثبت محاسبه شده است در نظر گرفته می‌شود. تنها مرسوله‌هایی که وضعیت آن‌ها تحویل داده شده است در محاسبه لحاظ می‌شوند. در گزارش علاوه بر مقدار کل درآمد، لیست تمام مرسوله‌های تحویل‌شده همراه با هزینه آن‌ها نیز چاپ می‌شود. شناسه‌ی مرسوله‌ها باید به ترتیب صعودی در خروجی چاپ شوند.

قالب ورودی
financial_report
قالب خروجی
Total income: <sum> Delivered orders: <order_id1> <cost1> <order_id2> <cost2> ...

نمونه ورودی
financial_report
نمونه خروجی
Total income: 50000 Delivered orders: 1 25000 3 15000 5 10000

نکات و نحوه تحویل

- برنامه شما باید در زمان معقول برای ورودی های آزمون اجرا شود.
- تحویل این تمرین در سامانه ای‌لرن انجام می‌شود. کد خود را در فایلی به نام A3-<SID>.cpp قرار دهید که در نام آن SID شماره دانشجویی شماست. سپس این فایل را در محل مشخص شده در ای‌لرن بارگذاری کنید. برای مثال اگر شماره دانشجویی شما 810100000 باشد، نام فایل بارگذاری شده توسط شما باید A3-810100000.cpp باشد. دقت داشته باشید که از کاراکتر خط فاصله[2] (-) استفاده کنید و از خط زیرین[3] (_) استفاده نکنید.
- دقت کنید که عدم رعایت ساختار گفته شده در بارگذاری باعث کسر ۱۰ درصد از نمره شما خواهد شد.
- **بخش مهمی از ارزیابی برنامه شما درستی عملکرد آن است. بنابراین به اندازه کافی ورودی‌های آزمایشی طراحی کنید تا درستی خروجی در حالت‌های مختلف آزموده شود و دقت کنید که اگر برنامه شما با مثال‌های داده شده درست کار کند لزوما به معنای درستی در تمام سناریوها نیست.**
- درستی برنامه شما از طریق آزمون‌های خودکار سنجیده می‌شود. به این ترتیب، لازم است خروجی تولید شده از نظر بزرگی و کوچکی حروف، رعایت فاصله‌ها، عدم وجود خروجی‌های اضافه، ... دقیقا مانند نمونه‌های داده شده باشد. بنابراین پیشنهاد می‌شود که با استفاده از ابزارهایی مانند diff فرمت خروجی برنامه خود را با خروجی‌هایی که در اختیارتان قرار داده شده است مطابقت دهید.
- هدف این تمرین یادگیری شماست. لطفا تمرین را خودتان انجام دهید. در صورت کشف تقلب مطابق سیاست درس با آن برخورد خواهد شد.

---

[2] Hyphen

[3] Underscore

نمرات

- تمیزی کد
  - رعایت کردن نام‌گذاری صحیح و منسجم بودن[4]
  - عدم وجود کد تکراری
  - رعایت دندانه‌گذاری[5]
  - استفاده **صحیح** از متغیرهای ثابت[6] به جای Magic Value-ها[7]
  - استفاده مناسب از container-ها و iterator-ها به جای روش‌های قدیمی
  - استفاده مناسب از ساختار داده‌های مناسب
  - توابع کوتاه که فقط یک کار را انجام می‌دهند
- درستی کد
  - آزمون‌های خودکار
- طراحی
  - شکستن مناسب و مرحله به مرحله مسئله
  - ذخیره اطلاعات در ساختار داده‌های مناسب
  - ساختاردهی کد در قالب توابع کوتاه که فقط یک کار را انجام می‌دهند
  - پرهیز از متغیرهای سراسری

دقت کنید که موارد ذکر شده لزوما کل نمره شما را تشکیل نمی‌دهند و ممکن است با تغییراتی همراه باشند.

---

[4] Consistency

[5] Indentation

[6] Constant

[7] به مقادیر خاصی که در کد استفاده می‌شود و برای عملکرد صحیح کد ضروری است اما دلیل استفاده از آن‌ها مشخص نیست و قابل جایگزین شدن با یک ثابت با اسم مشخص جهت افزایش خوانایی هستند، magic value گفته می‌شود. برای آشنایی بیشتر با این مفهوم می‌توانید این لینک را مشاهده کنید.